

In re Application of:

Thomas E. Saulpaugh, et al.

Serial No. 09/692,765

Filed: October 19, 2000

For: Event Message Endpoints in a Distributed Computing Environment

§ Group Art Unit: 2154

§ Examiner: Patel, Ashokkumar B.

§ Atty. Dkt. No.: 5181-65700
§ P5014

CERTIFICATE OF MAILING
37 C.F.R. § 1.8

I hereby certify that this correspondence is being deposited with the U.S. Postal Service with sufficient postage as First Class Mail in an envelope addressed to: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450, on the date indicated below:

Robert C. Kowert

Name of Registered Representative

November 16, 2005
Date

Signature

APPEAL BRIEF

Mail Stop Appeal Brief - Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir/Madam:

Further to the Notice of Appeal filed August 25, 2005, Appellants present this Appeal Brief. Appellants respectfully request that the Board of Patent Appeals and Interferences consider this appeal.

11/22/2005 EFLORES 00000077 501505 09692765

01 FC:1402 500.00 DA

I. REAL PARTY IN INTEREST

As evidenced by the assignment recorded at Reel/Frame 011478/0218, the subject application is owned by Sun Microsystems, Inc., a corporation organized and existing under and by virtue of the laws of the State of Delaware, and now having its principal place of business at 4150 Network Circle, Santa Clara, CA 95054.

II. RELATED APPEALS AND INTERFERENCES

No other appeals, interferences or judicial proceedings are known which would be related to, directly affect or be directly affected by or have a bearing on the Board's decision in this appeal.

III. STATUS OF CLAIMS

Claims 1-48 stand finally rejected. The rejection of claims 1-48 is being appealed. A copy of claims 1-48 is included in the Claims Appendix herein below.

IV. STATUS OF AMENDMENTS

No amendments to the claims have been submitted subsequent to the final rejection.

V. SUMMARY OF CLAIMED SUBJECT MATTER

Independent claim 1 is directed to a method for handling events in a distributed computing environment including receiving a message in a data representation language sent to a client platform in the distributed computing environment from a service in the distributed computing environment. The message includes a data representation language representation of an event generated by the service. As described in Appellants' description, some embodiments of a distributed computing environment may support an event "publish and subscribe" messaging model using event message endpoints (also referred to as event message gates). A message gate may be a message

endpoint for a client in a distributed computing environment to receive event notification messages in a “publish and subscribe” event messaging model in which services publish events to subscribing processes, sometimes referred to as clients. In some embodiments, message gates are used to both send and receive the published events. In some embodiments, a message gate may provide a secure message endpoint that sends and receives type-safe messages. Messages may be in a data representation language, such as XML in some embodiments. Thus, message gates may allow clients and services to exchange data representation language messages in a secure and reliable fashion over any suitable message transport, such as HTTP in some embodiments. *See, e.g.*, FIG. 2 - 5, 7-12, 14, 44, 45; page 12, lines 4-15; page 27, line 18 – page 28, line 29; page 29, lines 1-28; page 30, line 1-21; page 52, lines 16 – 22; page 53, lines 1- 25.

Message gates may also support publish and subscribe message passing for events. Message gates with event support may be referred to as event gates. In some embodiments, an event message endpoint (or gate) may recognize a set of events published by a service, subscribe to events, receive event messages generated by the service, and distribute the received events to processes, such as clients, that have registered interest in the events with the event message endpoint. When an event is generated by a service, a message including a data representation language representation of the event may be sent to each event message endpoint subscribed as a consumer of the event. After receiving an event message from a service, an event message endpoint may extract the data representation language representation of the event from the message and distribute the event. An event gate (also referred to as an event message endpoint) may be constructed from the XML schema indicating a set of one or more events that may be published by the service. An event gate may be configured to recognize some or all of the set of events published by a service, subscribe to those events, and distribute each event as the event is produced by the service. The event gate may subscribe by sending a subscription message for each event to which the gate desires to be subscribed. *See, e.g.*, FIG. 2 - 5, 7-12, 14, 15, 44 and 45; page 31, lines 13-25; line 27 – page 32, line 13; page 32, line 21 – page 33, line 23; page 33, line 24 – page 34, line 19; page 52, lines 16-29.

The method of claim 1 also includes sending the data representation language representation of the event to processes registered to receive the event from the service. For example, event consumers, such as clients, or other processes may subscribe with the event message endpoint for various types of events. In some embodiments, an event consumer may supply an event handler callback method to the event message endpoint. As event messages arrive at the event message endpoint, the event message endpoint may call each event handler method, thus passing the data representation language representation of the event to each subscriber. Thus, event message endpoints may distribute data representation language representations of events sent in messages from services to processes registered to receive the events. *See, e.g.*, FIG. 44, 45; page 13, lines 9-22; page 52, lines 16 – 22; page 53, lines 1- 25; page 54, line 16 – page 55, line 24.

Independent claim 14 is directed to a device including a processor, memory that is coupled to the processor and an event message gate unit. The message gate unit is configured to receive a message in a data representation language sent to a device in a distributed computing environment from a service in the distributed computing environment. The message includes a data representation language representation of an event generated by the service. As described above regarding claim 1, A message gate may is a message endpoint for a client in a distributed computing environment to receive event notification messages in a “publish and subscribe” event messaging model. A message gate may provide a secure message endpoint that sends and receives type-safe messages. Messages may be in a data representation language, such as XML in some embodiments. Thus, message gates may allow clients and services to exchange data representation language messages in a secure and reliable fashion over any suitable message transport, such as HTTP in some embodiments. *See, e.g.*, FIG. 2 - 5, 7-12, 14, 44, 45; page 12, lines 4-15; page 27, line 18 – page 28, line 29; page 29, lines 1- 28; page 30, line 1-21; page 52, lines 16 – 22; page 53, lines 1- 25;

The event message gate unit of claim 14 is also configured to send the data representation language representation of the event to processes registered to receive the

event from the service. As with the method of claim 1, described above, an event consumer may supply an event handler callback method to the event message endpoint. As event messages arrive at the event message endpoint, the event message endpoint may call each event handler method, thus passing the data representation language representation of the event to each subscriber. *See, e.g.*, FIG. 44, 45; page 13, lines 9-22; page 52, lines 16 – 22; page 53, lines 1- 25; page 54, line 16 – page 55, line 24.

Independent claim 27 is directed to a device including a processor, memory, and a service process configured to generate an event and generate a message in a data representation language, where the message includes a data representation language representation of the event. The service process of claim 27 is also configured to send the message to event message gate units in a distributed computing environment. As described above, services may generate and publish events using data representation language representations of events included in messages also in a data representation language. Please refer to the discussion of claim 1 above for a more detailed description of including data representation language representations of events in messages in a data representation language.

Claim 27 also includes where each of the event message gate units is operable to distribute the data representation language representation of the event sent in the message from the service process to processes registered to receive the event from the service process. Please refer to the discussion of claim 1 above for a more detailed description of event message gate units distributing data representation language representations of events to processes registered to receive the events.

Independent claim 36 is directed to a tangible, computer accessible medium including program instructions that are computer executable to implement the method of claim 1, described above. Please refer to the discussion of claim 1 above for a more detailed description.

The Summary above describes various examples and embodiments of the claimed

subject matter; however, the claims are not necessarily limited to any of these examples and embodiments. The claims should be interpreted based on the wording of the respective claims.

VI. GROUND OF REJECTION TO BE REVIEWED ON APPEAL

1. Claims 1-11, 14-24, 27-33 and 36-46 stand finally rejected under 35 U.S.C. § 102(e) as being anticipated by Bass et al. (U.S. Patent 6,549,956) (hereinafter “Bass”).

2. Claims 12, 13, 25, 26, 34, 35, 47 and 48 stand finally rejected under 35 U.S.C. § 103(a) as being anticipated by Bass in view of Meltzer et al. (U.S. Patent 6,542,912) (hereinafter “Meltzer”).

VII. ARGUMENT

First Ground of Rejection:

Claims 1-11, 14-24, 27-33 and 36-46 stand finally rejected under 35 U.S.C. § 102(e) as being anticipated by Bass et al. (U.S. Patent 6,549,956) (hereinafter “Bass”). Appellants traverse this rejection for at least the following reasons. Different groups of claims are addressed under their respective subheadings.

Claims 1, 11, 14, 24, 36 and 46:

Regarding claim 1, contrary to the Examiner’s assertion, Bass clearly fails to disclose receiving a message in a data representation language sent to a client platform in the distributed computing environment from a service in the distributed computing environment, wherein the message includes a data representation language representation of an event generated by the service. Bass teaches a mechanism for connecting disparate publication and subscribe domains via the Internet in which two channel adapters together act as a bridge across the network. Specifically, Bass teaches the use of existing network protocols (SMTP, TCP/IP, etc) via existing

holes in firewalls (column 2, lines 32-34). The Examiner cites portions of Bass (col. 2, lines 4-9, and 15-31, and column 3, lines 43-50) that describe how Bass' channel adapters receive published events, translate them into a message format suitable for transmission over a network, and send them to a channel adapter on another platform that translates the message back into the original event information.

The Examiner contends that by disclosing the translation of event information into network protocol messages, Bass discloses "that an event (message) can be represented in any data representation language and will be converted back into the event format for use in the other domain" (Office Action, page 3, lines 4-6). The Examiner further argues that by stating that channel adapters can convert event information into a format acceptable by the network, Bass discloses, "that an event (message) can be represented in any data representation language." However, Examiner reads too much into the actual teachings of Bass. The Examiner is arguing that the phrase "a format acceptable by the network" discloses the use of any data representation language. However, a data representation language is a particular type of language. As is well understood by anyone of ordinary skill in the art, a data representation language (such as XML) has a particular structure as a language and is a language for representing (or describing) data or content. There is clearly no teaching in Bass that any messages are sent in a particular data representation language. Nor is there any teaching in Bass the events are represented in such a language. Without some clear teaching by Bass regarding the use of a data representation language, Bass cannot be said to anticipate a message in a data representation language including a data representation language representation of an event. The Examiner's hindsight-based speculation regarding the possible use of a data representation language in Bass for messages is clearly improper in a rejection based on anticipation under 35 U.S.C. § 102(e).

In response to the above argument, the Examiner, in the Response to Argument and in the Advisory Action, refers to Bass' teachings regarding sending event information in an email via SMTP. Apparently the Examiner is arguing that SMTP is a data representation language. However, SMTP is a protocol, not a data representation

language. Data representation languages are well understood in the art. No one of ordinary skill in the art would consider SMTP (or any other similar network protocol) to be a data representation language. **The Examiner's reference to the use of SMTP supports Appellants' argument.** As is well understood by any one of ordinary skill in the art, the Simple Mail Transfer Protocol does not require any particular language, let alone a data representation language.

Bass teaches only the translation of event information into existing network protocol messages, such as an SMTP email, TCP/IP packet, or FTP transfer message. Bass does not teach that the messages sent using these protocols are messages in a data representation language. Bass teaches the use of existing network protocols in order to take advantage of the fact that existing network protocols use existing holes in firewalls and other security mechanisms (see, column 2, lines 15-35). For instance, Bass teaches that an event is formatted for transmission on a network (such as the Internet) and that “[t]he format may use transmission control protocol/Internet protocol (TCP/IP), simple mail transport protocol (SMTP), File Transfer Protocol (FTP), or whatever protocol is useable by the connecting network” (column 3, lines 35-42). Thus, Bass is describing the use of *protocols*, not any particular *language*. And Bass clearly does not mention using messages in a data representation language. Nor is there any reason to use messages in a data representation language in Bass's system, since none of the existing communications protocols advocated by Bass have any need or requirement for messages to be in a data representation language. Data representation languages are *specific types of languages* traditionally used in the prior art to describe documents or other content. XML is one example of a data representation language. Bass fails to mention anything about XML, **as admitted by the Examiner** regarding the rejection of claim 13, discussed below. As noted above, Bass fails to mention any data representation language. The prior art does not teach the use of a data representation language to represent events in messages between entities in a distributed computing environment.

In response to the above argument, the Examiner merely objects to Appellants' use of the phrase “data representation language messages” to refer to messages in a data

representation language, arguing that Appellants' claims do not recite the phrase "data representation language messages." However, the Examiner's objection is clearly misplaced as anyone of ordinary skill in the art would recognize that Appellants' previous use of the phrase, "data representation language messages" refers to the "a message in a data representation language" recited in Appellants' claim 1.

Additionally, Bass fails to anticipate that the message includes a data representation language representation of an event generated by the service. In contrast, Bass teaches channel adapters that "convert the event information into a format acceptable by the network" (column 2, lines 15-18). The "format acceptable by the network" in Bass is not described as a data representation language representation of an event. Bass clearly does not teach a message that includes a data representation language representation of an event. The Examiner cites only col. 2, lines 4-9, and 15-31 of Bass that, as noted above, describe how a channel adapter translates event information into network protocol messages. The Examiner does not cite any portion of Bass that refers to any data representation language representation of an event.

Furthermore, Bass fails to anticipate sending the data representation language representation of the event to one or more processes registered to receive the event from the service. The Examiner cites column 2, lines 9-15, where Bass describes a process adapter subscribing to and receiving an event via a channel adapter. However, Bass only teaches that the process adapter receives the event, not a data representation language representation of the event. The Examiner argues that by teaching how a channel adapter reformats an event for transmission over the Internet, Bass discloses the use of a data representation language representation of events. The Examiner's interpretation of Bass is incorrect. As noted above, Bass teaches only translating event information into a format suitable for transmission over the Internet via any of a number existing network protocols (such as TCP/IP, SMTP, FTP, etc). However, nowhere does Bass mention that the event information is a data representation language representation of an event.

In response the Appellants' argument above, the Examiner responds by citing

Bass column 3, lines 43-50 where Bass describes how channel adapters receive published events, translate them into a message format suitable for transmission over a network, and send them to a channel adapter on another platform that translates the message back into the original event information. Thus, when Bass' channel adapter delivers the event to the subscribing process, the channel adapter has already, "re-transformed" the email (used to send the event information) back into the event (See, Bass, column 3, lines 45-50). Bass specifically states that the channel adapter delivers the re-constituted event, *rather than any data representation language representation of the event*, to subscribing processes.

Appellants note that anticipation requires the presence in a single prior art reference disclosure of each and every element of the claimed invention, arranged as in the claim. *Lindemann Maschinenfabrik GmbH v. American Hoist & Derrick Co.*, 221 USPQ 481, 485 (Fed. Cir. 1984). The **identical** invention must be shown in as complete detail as is contained in the claims. *Richardson v. Suzuki Motor Co.*, 9 USPQ2d 1913, 1920 (Fed. Cir. 1989). Bass clearly fails to anticipate receiving a message in a data representation language sent to a client platform in the distributed computing environment from a service in the distributed computing environment, wherein the message includes a data representation language representation of an event generated by the service.

For at least the reasons given above, the rejection of claim 1 is not supported by the cited art and removal thereof is respectfully requested. Arguments similar to those presented above regarding claim 1 apply to claims 14 and 36 as well.

Claims 2, 7, 15, 21, 37 and 42:

Regarding claim 2, Bass fails to anticipate receiving a data representation language schema on the client platform, wherein the data representation language schema defines a message interface for a set of events generated by the service. The Examiner cites column 3, lines 43-50 that describes Bass' channel adapters. Bass teaches that each

channel adapter includes two interfaces, a framework interface and a protocol interface (column 3, lines 53-64). The framework interface includes domain specific protocols for communicating published and subscribed events with a domain broker. The protocol interface includes network specific protocols that enable the adapter to couple with the Internet. The Examiner has not cited any portion of Bass that teaches a *data representation language schema* defining a message interface for a set of events. Instead, Basses Bass teaches that each channel adapter includes two different interfaces for communicating event information.

The Examiner also cites column 2, lines 4-15 and column 4, line 43 – column 5, line 15 where Bass teaches how each of his channel adapters are configured with a set of events it will export to a peer adapter in another domain. Bass teaches how a system administrator configures each channel adapter to receive and transmit specific events and how channel adapters exchange, or export, lists of events that they will be communicating. The Examiner argues that exchanging event lists amounts to receiving a data representation language schema defining a message interface for a set of events. However, Bass' event list exchange only informs the channel adapter which events will be communicated. Bass does not teach that his event export lists make up a *data representation language schema*. Bass also does not mention that the event export lists are exchanged using a data representation language. Furthermore, Bass does not describe his event export lists as defining message interfaces. To the contrary, as discussed above, Bass teaches (column 3, lines 53-64) that each channel adapter includes a protocol interface that includes network protocol messages. A channel adapter's protocol interface has nothing to do with the list of events that it may send and receive. Bass teaches that the channel adapter can convert any event into an appropriate network protocol messages. Thus, the exchange of exported event lists cited by the Examiner does not teach anything regarding receiving a data representation language schema defining a message interface.

Additionally, Bass does not teach generating an event message endpoint for the client platform according to the data representation language schema. The Examiner

cites Bass' teachings regarding the receiving of events listed on an event type list (column 4, line 43 – column 5, line 15) and argues that Bass discloses generating an event message endpoint according to a data representation language schema by describing how an event on an event type list is received and re-published via the channel adapter. The Examiner's interpretation of Bass is clearly incorrect. As discussed above, Bass' exported event type lists are not data representation language schemas. Moreover, not only do the event type lists in Bass not involve the generation of any message endpoints, they also have absolutely nothing to do with a data representation language schema. Thus, Bass clearly fails to disclose generating an event message endpoint for the client platform according to the data representation language schema.

In the Response to Arguments and the Advisory Action, the Examiner responds to the arguments above by merely repeating the rejection of claim 2. The Examiner does not provide any additional explanation or argument regarding Appellants' arguments. Thus, the Examiner has not provided any actual rebuttal to Appellants' arguments.

Thus, for at least the reasons give above, the Examiner's rejection of claim 2 is not supported by the prior art and removal thereof is respectfully requested. Similar arguments as those presented above apply to claims 15 and 37 as well.

Claims 3, 19 and 38:

Regarding claim 3, Bass fails to disclose the event message endpoint subscribing to one or more of the set of events generated by the service, wherein the service is configured to send messages including data representation language representations of an event to subscribers to the event when the event is generated. The Examiner cites column 3, lines 43-50 of Bass describing how an event is delivered to a subscribing channel adapter. However, as discussed above regarding claims 1 and 2, Bass fails to teach anything regarding a service configured to send messages including *data representation language representations of events*. Instead, Bass teaches that events are converted into network protocol messages, such as SMTP email messages, for transmission over the

Internet where they are converted back into the original event information for re-publishing in a different domain. However, as described above regarding claims 1 and 2, such network protocols are not data representation languages. Bass does not describe these protocol messages as being messages in a data representation language. Furthermore, nowhere does Bass mention a service configured to send messages including data representation language representations of an event.

In response to the above arguments, the Examiner, in the Response to Arguments, cites column 3, lines 49-50 where Bass states that channel adapters then delivers the event to any subscribing process adapters within the domain. However, as noted above, prior to delivering the event to the subscribing process adapters, the channel adapter converts the event information from the network protocol message (e.g. SMTP email message) back into the event. (see, Bass, column 3, lines 45-50). The event delivered by the channel adapters is clearly not a data representation language representation of the event. Thus, the Examiner's cited passage actually supports Appellants' argument.

For at least the reasons above, the rejection of claim 3 is not supported by the prior art and removal thereof is respectfully requested. Similar arguments as those presented above apply to claims 19 and 38 as well.

Claims 4, 20 and 39:

In regards to claim 4, Bass fails to disclose wherein the data representation language message from the service includes an authentication credential for the service. Bass additionally fails to disclose the event message endpoint using the authentication credential for the service to authenticate the data representation language message as being from the service. The Examiner cites column 4, line 57 to column 5, line 15 of Bass that describes how Bass' channel adapters are configured to send and receive various events. Please see the discussion above regarding claim 2 for a more detailed discussion of this portion of Bass. The Examiner does not provide any argument or discussion regarding how Bass' exported event type lists have relevance to a data

representation language message including an authentication credential. Nowhere does Bass mention anything regarding data representation language messages including authentication credentials nor about event message endpoints using an authentication credential to authenticate the data representation language message.

In the Response to Arguments, the Examiner cites column 1, lines 56-60 of Bass and argues that Bass' stated purpose for his invention, namely, "there is a need in the art for a mechanism to link to disparate PUB/SUB domains together without compromising security, reducing performance, be easy to implement, and still allow for information transfer between the two domains" discloses the specific limitations of Appellants' claim 4. However, a general statement regarding Bass intention that his invention no compromise security does not disclose or anticipate the specific limitation of a data representation language message from a service including an authentication credential for the service. Nor does the cited passage anticipate an event message endpoint using the authentication credential for the service to authenticate the data representation language message as being from the service.

As noted above regarding claim 1, anticipation requires the presence in a single prior art reference disclosure of each and every element of the claimed invention, arranged as in the claim. The passages cited by the Examiner can in no way be considered to disclose each and every element of Appellants' claim 4, arranged as in the claim. Thus, for at least the reasons above, the rejection of claim 4 is not supported by the prior art and removal thereof is respectfully requested. Similar arguments as those presented above apply to claims 20 and 39 as well.

Claims 5, 16 and 40:

Regarding claim 5, Bass fails to disclose the event message endpoint verifying type correctness of the data representation language message according to the data representation language schema. The Examiner cites column 2, lines 24-27 and column 3, lines 45-50 of Bass. However, the first cited portion only describes how Bass' channel

adapters use a plurality of states and status messages to track and indicate the delivery, receipt, and publication of events. The second cited portion describes how an event is transformed into an email message via SMTP and then re-transformed back into the event upon receipt.

Neither of the Examiner's cited portions of Bass has anything to do with *verifying type correctness* of a data representation language message *according to a data representation language schema*. The various states and status messages indicating delivery, receipt, and publication of events only track and help to guarantee that event messages are eventually delivered to the subscribing process adapter. These states have nothing to do with verifying type correctness of a data representation language message according to a data representation language schema. Similarly, converting an event into an email message via SMTP (or another network protocol message) and converting the message back into an event has nothing to do with verifying type correctness of a data representation language message according to a data representation language schema. In fact, nowhere does Bass make any reference whatsoever to verifying type correctness of a data representation language message according to a data representation language schema.

Furthermore, the Examiner has argued that Bass' exported event type lists constitute a data representation language schema (see, Final Office Action, pages 14-15, regarding claim 2). However, Bass does not teach that an exported event type list has anything do with the states and status messages indicating delivery, receipt, and publication of events or have anything to do with converting events into SMTP messages. Thus, the Examiner's interpretation of Bass is inconsistent and thus cannot be correct.

In the Response to Arguments, the Examiner responds to the above arguments by citing column 4, lines 18-24 where Bass describes how each channel adapter includes a reporting mechanism. Bass describes this reporting mechanism has informing an administrator of the status of events and that the administrator can "determine if there are any events that are stuck, and the state in which they are stuck." However, reporting on

the state of events as they flow through Bass' system does not disclose the specific functionality of verify type correctness of a data representation language message according to a data representation language schema. Not only does the cited passage fail to mention verifying type correctness of any messages, the passage also fails to mention anything regarding the use of a data representation language schema to verify type correctness.

In the Advisory Action, the Examiner response to the above argument by repeating the same response from the Response to Argument section of the Final Office Action and the same citation from Bass (column 4, lines 18-24). However, as noted above, this passage of Bass has no relevance to verifying the type correctness of messages using a data representation language schema. The Examiner has not provided any additional arguments regarding how Bass' discussion of stuck messages involves verifying type correctness. Nor has the Examiner cited any additional portion of Bass that mentions using a data representation language schema to verify type correctness.

Additionally, the reporting mechanism relied on by the Examiner (in the Response to Arguments and the Advisory Action) is not Bass' Channel Adapter, which the Examiner has previously equated to the event message endpoint of Appellants' claims (see Examiner's rejection of claim 2, Final Office Action, page 14, lines 4-10). Appellants' claim 5 recites the event message endpoint verifying type correctness of the message according to a data representation language schema. Even if Bass' Channel Adapter were to verify type correctness of message according to a data representation language schema, which it does not, Bass still fails to anticipate Appellants' claim 5.

The Examiner's has clearly failed to provide a proper rejection based on anticipation. The rejection of claim 5 is not supported by the prior art and removal thereof is respectfully requested. Similar arguments as those presented above apply to claims 16 and 40 as well.

Claims 6, 17 and 41:

Regarding claim 6, Bass fails to anticipate wherein the data representation language schema defines a set of messages that the service may send to the event message endpoint and further fails to teach the event message endpoint verifying the correctness of the data representation language message from the service according to the data representation language schema. The Examiner once again cites column 2, lines 24-27 and column 3, lines 45-50 of Bass. However, as noted above regarding claim 5, the first cited portion only describes how Bass' channel adapters use a plurality of states and status messages to indicate the delivery, receipt, and publication of events and the second cited portion describes how an event is transformed into an email message via SMTP and then re-transformed back into the event upon receipt. As discussed above regarding claim 5 (for which the Examiner cites the same portions of Bass), neither of the Examiner's cited portions have anything to do with a data representation language schema defining a set of messages that a service may send to an event message endpoint. Additionally, neither of the cited passages mentions an event message endpoint verifying the correctness of a data representation language message according to the data representation language schema.

Specifically, the various states and status messages (indicating delivery, receipt, and publication of events) only help to track and guarantee that event messages are eventually received by the subscribing process adapter. These states have nothing to do with verifying the correctness of a data representation language message according to a data representation language schema. Similarly, converting an event into an email message via SMTP (or another network protocol message) and converting the message back into an event is not verifying type correctness of a data representation language message according to a data representation language schema.

Thus, the rejection of claim 6 is clearly not supported by the prior art and removal thereof is respectfully requested. Similar arguments as those presented above apply to claims 17 and 41 as well.

Claims 8, 22 and 43:

Regarding claim 8, Bass fails to disclose each of the one or more processes providing an event handler callback method to the event message endpoint. The Examiner cites column 4, lines 57-60. However, this portion of Bass only teaches that Bass' channel adapters subscribe to and publish events, but fails to describe any mechanism for delivering the events other than via existing network protocols. Nowhere does Bass teach providing an event handler callback method to an event message endpoint.

In response to the arguments above, the Examiner cites column 4, line 43 through column 5, line 16 and refers to Bass' teaching that channel adapters republish events to interested process adapters. Specifically the Examiner refers to Bass teaching how "prior to transfer of events between the domains, the respective process and channel adapters of the domains must be configured to send and receive the different events" (Bass, column 4, lines 57-59). However, merely stating that the process and channel adapters must be configured to send and receive events in no way discloses, teaches, or even implies providing an event handler callback method to an event message endpoint. Without some clear and specific teaching by Bass regarding providing an event handler callback method, the Examiner is merely speculating as to the details of Bass system.

Bass further fails to teach the event message endpoint calling an event handler method of each process registered with the event message endpoint and the event message endpoint passing the data representation language representation of the event to each called event handler. The Examiner cites column 3, lines 22-50 where Bass describes how his channel adapters convert events to and from network protocol message and how events are sent over the Internet and re-published in other domains. However, nowhere does Bass describe an event message endpoint calling an event handler method. Nor does Bass teach an event message endpoint passing a data representation language representation of an event to each called event handler. The Examiner merely states, "the

reference teaches that the processes as well as the adapters are configured to do the claimed element” and further claims, “the c[h]annel adapters are capable of executing the task as claimed.” However, without any supporting teaching from Bass, the Examiner’s rejection amounts to nothing more than mere hindsight speculation and conclusory statements.

Thus, for at least the reasons given above, the rejection of claim 8 is not supported by the prior art and removal thereof is respectfully requested. Similar arguments as those presented above apply to claims 22 and 43 as well.

Claims 9, 23 and 44:

Regarding claim 9, Bass fails to disclose, contrary to the Examiner’s assertion, a process unregistering interest in a first event of the service. The Examiner cites column 4, line 57 through column 5, line 15 of Bass. However, this passage of Bass only refers delivering events to channel adapters that have subscribed to receive the event. The passage cited by the Examiner does not mention anything about a process unregistering interest in an event. In fact, nowhere does Bass mention a channel adapter, or other process, unsubscribing to events.

Additionally, Bass fails to the event message gate unsubscribing to the event with the service subsequent to the unregistering. The Examiner fails to cite any portion of Bass describing an event message gate unsubscribing to an event. The Examiner’s cited passage, described above, fails to mention anything about an event message gate, or any other component of Bass’ system, unsubscribing to events. Instead, as noted above, the cited passage only refers to delivering events to subscribing channel adapters. As noted above, Bass fails to teach anything about unsubscribing to events.

Thus, the rejection of claim 9 is not supported by the prior art and removal thereof is respectfully requested. Similar arguments as those presented above apply to claims 23 and 44 as well.

Claims 10, 18 and 45:

Regarding claim 10, Bass does not disclose receiving the data representation language schema of the service in a service advertisement of the service. The Examiner cites column 2, lines 4-15, column 3, lines 43-50 and column 4, line 43 – column 5, line 15, where Bass describes how his channel adapters are configured with a set of events it will export to its peer adapter in another domain. Please refer to the remarks above regarding claim 2 for a discussion of these portions of Bass. The Examiner apparently contends that Bass's use of exported event type lists include service advertisements. However, the exported event type lists have absolutely nothing to do with a service advertisement that includes a data representation language schema.

In the Response to Arguments, the Examiner again cites column 2, lines 4-15, column 3, lines 43-50 and column 4, line 43 – column 5, line 15 of Bass, without providing any additional argument or interpretation regarding Appellants' argument above.

For at least the reasons given above, the rejection of claim 10 is not supported by the prior art and removal thereof is respectfully requested. Similar arguments as those presented above apply to claims 18 and 45 as well.

Claims 27 and 28:

Regarding claim 27, Bass fails to anticipate a service process configured to generate a message in a data representation language. The Examiner cites column 2, lines 49, and 15-31 of Bass and argues that converting event information into a format acceptable by the network discloses that an event (message) can be represented in any data representation language. The Examiner's interpretation of Bass is incorrect. As discussed above regarding claim 1, Bass teaches the use of existing network protocols such as SMTP, TCP/IP, or FTP which have absolutely no bearing whatsoever on the use of a data representation language. Bass does not mention anything regarding using data

representation language messages.

Bass further fails to anticipate wherein the message includes a data representation language representation of the event generated by the service process. The Examiner does not cite any passage in Bass that refers to a message including a data representation language representation of an event, as suggested by the Examiner. Instead, Bass teaches that the event information is translated into a network protocol message, as described above regarding claim 1. Furthermore, as defined in the art, existing network protocols do not include data representation language representation of events.

Bass also does not anticipate wherein each of the one or more event message gate units is operable to distribute the data representation language representation of the event, as asserted by the Examiner. Also as noted above regarding claim 1, Bass teaches that once received by a channel adapter, the network protocol message is converted back into the original event information. Thus, in order to distribute data representation language representations of an event, an event would have to originally be a data representation language representation of the event. However, Bass does not teach anything regarding data representation language representations of events. The Examiner has not cited any passage of Bass that refers to data representation language representations of an event.

In the Response to Arguments, the Examiner cites elements 16 and 17 of Bass' FIG. 1 and refers to the previous Response to Arguments regarding claim 1. However, FIG. 1 of Bass does not illustrate a message in a data representation language or data representation language representations of events generated by the service process. Furthermore, as noted above, Bass teaches the use of existing network protocols, such as SMTP, TCP/IP, or FTP, which, as discussed above, are not data representation languages.

For at least the reasons given above, the rejection of claim 27 is not supported by the prior art and removal thereof is respectfully requested.

Claim 29:

Regarding claim 29, Bass fails to anticipate a service process configured to provide a data representation language schema defining a message interface for a set of events generated by the service and also fails to teach wherein one or more event message gate units are generated according to the data representation language schema. The Examiner cites column 3, lines 43-50 that describes Bass' channel adapters. Bass teaches that each channel adapter includes two interfaces, a framework interface and a protocol interface (column 3, lines 53-64). The framework interface includes domain specific protocols for communicating published and subscribed events with a domain broker. The protocol interface includes network specific protocols that enable the adapter to couple with the Internet. The Examiner has not cited any portion of Bass that teaches a data representation language schema defining a message interface for a set of events. Instead, Bass teaches that each channel adapter includes two different interfaces for communicating event information.

The Examiner also cites and column 2, lines 4-15 and column 4, line 43 – column 5, line 15 where Bass teaches how his channel adapters are configured with a set of events it will export to its peer adapter in another domain. Bass teaches how an administrator configured each channel adapter to receive and transmit specific events and how channel adapters exchange lists of events that they will be communicating. The Examiner argues that exchanging event lists amounts to receiving a data representation language schema defining a message interface for a set of events. However, Bass' event list exchange only informs the channel adapter which events will be communicated. Bass does not teach that his event export lists are data representation language schemas. Bass does not mention that the event export lists are exchanged using a data representation language. Furthermore, Bass does not describe his event export lists as defining a message interfaces. On the contrary, as discussed above, Bass teaches (column 3, lines 53-64) that each channel adapter includes a protocol interface that includes network protocol messages. A channel adapter's protocol interface has nothing to do with the list of events that it may send and receive. Bass teaches that the channel adapter can convert

any event into an appropriate network protocol messages. Thus, the exchange of exported event lists cited by the Examiner does not teach anything regarding receiving a data representation language schema defining a message interface.

Bass also fails to teach generating event message gate units according to a data representation language schema. The Examiner cites Bass' teachings regarding the receiving of events listed on an event type list (column 4, line 43 – column 5, line 15) and argues that Bass discloses generating event message gate units according to a data representation language schema by describing how an event on an event type list is received and re-published via the channel adapter. The Examiner's interpretation of Bass is clearly incorrect. As discussed above, Bass' exported event type lists are not data representation language schemas. Not only do the event type lists fail to define any message interfaces, they also do not use a data representation language.

Thus, for at least the reasons given above, the Examiner's rejection of claim 29 is not supported by the prior art and removal thereof is respectfully requested.

Claim 30:

Regarding claim 30, Bass fails to anticipate wherein the data representation language schema defines a set of messages that the service may send to the event message gate units. The Examiner cites column 2, lines 24-27 and column 3, lines 45-50 of Bass. However, as noted above regarding claims 5 and 6, the first cited portion only describes how Bass' channel adapters use a plurality of states and status messages to indicate the delivery, receipt, and publication of events and the second cited portion describes how an event is transformed into an email message via SMTP and then re-transformed back into the event upon receipt. As discussed above regarding claim 5 (for which the Examiner cites the same portions of Bass), neither of the Examiner's cited portions have anything to do with a data representation language schema defining a set of messages that a service may send to an event message gate units. Bass is silent regarding a data representation language schema defining a set of messages that a service

may send to the event message gate units. Thus, the rejection of claim 30 is clearly not supported by the prior art and removal thereof is respectfully requested.

Claim 31:

Regarding claim 31, Bass does not teach a service process configured to provide the data representation language schema in a service advertisement. The Examiner cites column 2, lines 4-15, column 3, lines 43-50 and column 4, line 43 – column 5, line 15, where Bass describes how his channel adapters are configured with a set of events it will export to its peer adapter in another domain. Please refer to the remarks above regarding claim 2 for a discussion of these portions of Bass. The Examiner apparently contends that Bass use of exported event type lists include service advertisements. However, the exported event type lists have absolutely nothing to do with a service advertisement that includes a data representation language schema.

Thus, for at least the reasons given above, the rejection of claim 31 is not supported by the prior art and removal thereof is respectfully requested.

Claim 32:

Regarding claim 32, Bass fails to teach the event message endpoint subscribing to one or more of the set of events generated by the service, wherein the service is configured to send messages including data representation language representations of an event to subscribers to the event when the event is generated. The Examiner cites column 3, lines 43-50 of Bass describing how an event is delivered to a subscribing channel adapter. However, as discussed above, Bass fails to teach anything regarding a service configured to send message including data representation language representations of an event. Instead, Bass teaches that events are converted into network protocol messages for transmission over the internet where they are converted back into the original event information for re-publishing in a different domain. Nowhere does Bass mention a service sending messages including data representation language representations of an event.

Thus, the rejection of claim 32 is not supported by the prior art and removal thereof is respectfully requested.

Claim 33:

In regards to claim 33, Bass fails to disclose wherein the service process is further configured to attach an authentication credential for the service to the data representation language message, where the authentication credential is configured for use in authenticating the data representation language message as being from the service process. The Examiner cites column 4, line 57 to column 5, line 15 of Bass that describes how Bass' channel adapters are configured to send and receive various events. Please see the discussion above regarding claim 2 for a more detailed discussion of this portion of Bass. The Examiner does not provide any argument or discussion regarding how Bass' exported event type lists have relevance to a data representation language message including an authentication credential. Nowhere does Bass mention anything regarding data representation language messages including authentication credentials nor about event message endpoints using an authentication credential to authenticate the data representation language message.

In the Response to Arguments, the Examiner cites column 1, lines 56-60 of Bass and argues that Bass' stated purpose for his invention, namely, "there is a need in the art for a mechanism to link to disparate PUB/SUB domains together without compromising security, reducing performance, be easy to implement, and still allow for information transfer between the two domains" discloses the specific limitations of Appellants' claim 4. However, a general statement regarding Bass intention that his invention no compromise security does not disclose or anticipate the specific limitation of a data representation language message from a service including an authentication credential for the service. Nor does the cited passage anticipate an event message endpoint using the authentication credential for the service to authenticate the data representation language message as being from the service.

As noted above, anticipation requires the presence in a single prior art reference disclosure of each and every element of the claimed invention, arranged as in the claim. The passages cited by the Examiner can in no way be considered to disclose each and every element of Appellants' claim 33, arranged as in the claim. Thus, for at least the reasons above, the rejection of claim 33 is not supported by the prior art and removal thereof is respectfully requested.

Second Ground of Rejection:

Claims 12, 13, 25, 26, 34, 35, 47 and 48 stand finally rejected under 35 U.S.C. § 103(a) as being anticipated by Bass in view of Meltzer et al. (U.S. Patent 6,542,912) (hereinafter "Meltzer"). Appellants traverse this rejection for at least the following reasons. Different groups of claims are addressed under their respective subheadings.

Claim 12, 25 and 47:

Contrary to the Examiner's assertion, Bass in view of Meltzer fails to teach or suggest a message in a data representation language including a data representation language representation of an event, where the event is a JAVA event. The Examiner admits that Bass does not disclose the use of XML or JAVA events. The Examiner relies upon Meltzer, citing column 14, lines 25-32. Meltzer teaches a system for exchanging business related information using self-defining documents, such as XML based documents. The Examiner's cited passage describes turning an XML document into a set of Java events. However, Meltzer teaches that listeners in a publish and subscribe architecture do not have to understand XML, but instead may be JAVA based and Meltzer specifically teaches the translation of XML documents into JAVA events to allow listeners to processing received documents (Meltzer, column 14, lines 41-54). Thus, rather than teaching or suggesting using data representation language representations of JAVA events, as the Examiner contends, **Meltzer teaches the opposite**, using Java events to represent a data representation language (e.g. XML) document.

Thus, the combination of Bass and Meltzer would not result in a publish and subscribe system in which messages including data representation language representations of JAVA events, as recited in Appellants' claims. Instead, the Examiner's proposed combination of Bass and Meltzer would result in a publish and subscribe system in which data representation language documents are translated into Java events for listeners or other recipients of the documents.

Furthermore, since Bass fails to teach the use of XML or JAVA events, as the Examiner admits, it would not make sense, nor would it be obvious, to modify the system of Bass to translate XML documents into JAVA events as taught by Meltzer. The Examiner's stated motivation, namely, so that a transaction process front end is able to operate in a publish and subscribe architecture that enables the addition of new listener programs without the knowledge of or impact on other listening programs in the system" is merely a reason why someone would use Meltzer's system and does not provide any motivation to modify the teachings of Bass. As noted above, the Examiner is relying upon the teachings of Meltzer regarding translating XML documents to JAVA events. However, since Bass does not teach anything about either XML documents or JAVA events, there is no motivation to modify Bass' system to include these teachings of Meltzer.

Therefore, the rejection of claim 12 is not supported by the prior art and removal thereof is respectfully requested. Similar remarks also apply to claims 25 and 47.

Claim 13, 26 and 48:

Regarding claim 13, the Examiner's combination of Bass in view of Meltzer fails to teach or suggest receiving a message in a data representation language sent to a client platform in the distributed computing environment from a service in the distributed computing environment, wherein the message includes a data representation language representation of an event generated by the service, wherein the data representation

language is eXtensible Markup Language (XML). The Examiner admits that Bass fails to teach using XML and relies upon Meltzer, citing column 14, lines 25-32. As noted above regarding the rejection of claim 12, Meltzer teaches a system for exchanging business related information using self-defining documents, such as XML based documents. The Examiner's cited passage describes turning an XML document into a set of Java events. Thus, Meltzer does not teach or suggest sending messages in a XML, nor does Meltzer teach or suggest including an XML representation of an event in such a message. In fact, Meltzer teach away from including XML representations of events. Instead, as described above regarding claim 12, Meltzer teaches translating XML documents into JAVA events. Nowhere does Meltzer teach or suggest including an data representation language representation of an event in a message in a data representation language, where the data representation language is XML.

Since neither Bass nor Meltzer teach or suggest using XML for message or including an XML representation of an event in a message, the Examiner's combination of Bass and Meltzer also fails to teach or suggest sending messages in a XML, nor does Meltzer teach or suggest including an XML representation of an event in such a message. Instead, the Examiner's combination of Bass and Meltzer would result in the publish and subscribe system taught by Bass that also includes translating XML documents into JAVA events, as taught by Meltzer.

Thus, the rejection of claim 13 is not supported by the prior art and removal thereof is respectfully requested. Similar remarks apply to claims 26 and 48 as well.

Claim 34:

Contrary to the Examiner's assertion, Bass in view of Meltzer fails to teach or suggest a message in a data representation language including a data representation language representation of an event, where the event is a JAVA event. The Examiner admits that Bass does not disclose the use of XML or JAVA events. The Examiner relies upon Meltzer, citing column 14, lines 25-32. Meltzer teaches a system for exchanging

business related information using self-defining documents, such as XML based documents. The Examiner's cited passage describes turning an XML document into a set of Java events. However, as noted above regarding claim 12, Meltzer teaches that listeners in a publish and subscribe architecture do not have to understand XML, but instead may be JAVA based and Meltzer specifically teaches the translation of XML documents into JAVA events to allow listeners to processing received documents (Meltzer, column 14, lines 41-54). Thus, rather than teaching or suggesting using data representation language representations of JAVA events, as the Examiner contends, Meltzer teaches the opposite, using Java events to represent a data representation language (e.g. XML) document.

Thus, the combination of Bass and Meltzer would not result in a publish and subscribe system in which messages including data representation language representations of JAVA events, as recited in Appellants' claims. Instead, the Examiner's proposed combination of Bass and Meltzer would result in a publish and subscribe system in which data representation language documents are translated into Java events for listeners or other recipients of the documents.

Furthermore, since, as noted above regarding the rejection of claim 12, Bass fails to teach the use of XML or JAVA events, as the Examiner admits, it would not make sense, nor would it be obvious, to modify the system of Bass to translate XML documents into JAVA events as taught by Meltzer. The Examiner's stated motivation, namely, so that a transaction process front end is able to operate in a publish and subscribe architecture that enables the addition of new listener programs without the knowledge of or impact on other listening programs in the system" is merely a reason why someone would use Meltzer's system and does not provide any motivation to modify the teachings of Bass. As noted above, the Examiner is relying upon the teachings of Meltzer regarding translating XML documents to JAVA events. However, since Bass does not teach anything about either XML documents or JAVA events, there is no motivation to modify Bass' system to include these teachings of Meltzer.

Claim 35:

Regarding claim 35, the Examiner's combination of Bass in view of Meltzer fails to teach or suggest a service process configured to generate a message in a data representation language, wherein the message includes a data representation language representation of the event generated by the service process, wherein the data representation language is eXtensible Markup Language (XML). The Examiner admits that Bass fails to teach using XML and relies upon Meltzer, citing column 14, lines 25-32. As noted above regarding the rejection of claims 12 and 13, Meltzer teaches a system for exchanging business related information using self-defining documents, such as XML based documents. The Examiner's cited passage describes turning an XML document into a set of Java events. Thus, Meltzer does not teach or suggest sending messages in a XML, nor does Meltzer teach or suggest including an XML representation of an event in such a message. In fact, Meltzer teach away from including XML representations of events. Instead, as described above regarding claim 12, Meltzer teaches translating XML documents into JAVA events. Nowhere does Meltzer teach or suggest including an data representation language representation of an event in a message in a data representation language, where the data representation language is XML.

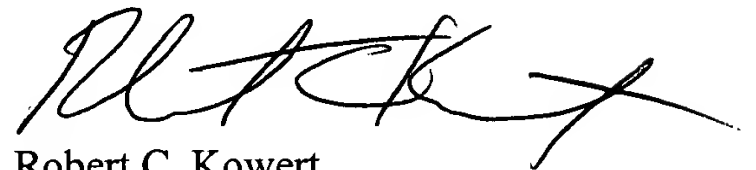
Since neither Bass nor Meltzer teach or suggest using XML for message or including an XML representation of an event in a message, the Examiner's combination of Bass and Meltzer also fails to teach or suggest sending messages in a XML, nor does Meltzer teach or suggest including an XML representation of an event in such a message. Instead, the Examiner's combination of Bass and Meltzer would result in the publish and subscribe system taught by Bass that also includes translating XML documents into JAVA events, as taught by Meltzer. No combination of Bass and Meltzer would include messages in a data representation language or data representation language representations of events, where the data representation language is XML. Thus, the rejection of claim 35 is not supported by the prior art and removal thereof is respectfully requested.

VIII. CONCLUSION

For the foregoing reasons, it is submitted that the Examiner's rejection of claims 1-48 was erroneous, and reversal of his decision is respectfully requested.

The Commissioner is authorized to charge the appeal brief fee of \$500.00 and any other fees that may be due to Meyertons, Hood, Kivlin, Kowert, & Goetzel, P.C. Deposit Account No. 501505/5181-65700/RCK. This Appeal Brief is submitted with a return receipt postcard.

Respectfully submitted,

A handwritten signature in black ink, appearing to read 'R. C. Kowert', with a stylized flourish at the end.

Robert C. Kowert
Reg. No. 39,255
Attorney for Appellants

Meyertons, Hood, Kivlin, Kowert & Goetzel, P.C.
P.O. Box 398
Austin, TX 78767-0398
(512) 853-8850

Date: November 16, 2005

IX. CLAIMS APPENDIX

The claims on appeal are as follows.

1. A method for handling events in a distributed computing environment, comprising:

receiving a message in a data representation language sent to a client platform in the distributed computing environment from a service in the distributed computing environment, wherein the message includes a data representation language representation of an event generated by the service; and

sending the data representation language representation of the event to one or more processes registered to receive the event from the service.

2. The method as recited in claim 1, further comprising:

receiving a data representation language schema on the client platform, wherein said data representation language schema defines a message interface for a set of events generated by the service; and

generating an event message endpoint for the client platform according to the data representation language schema, wherein said receiving a message and said sending the data representation language representation of the event to one or more processes are performed by the event message endpoint.

3. The method as recited in claim 2, further comprising the event message endpoint subscribing to one or more of the set of events generated by the service, wherein the service is configured to send messages including data representation language representations of an event to subscribers to the event when the event is generated.

4. The method as recited in claim 2, wherein the data representation language message from the service includes an authentication credential for the service, the method further comprising the event message endpoint using the authentication credential for the service to authenticate the data representation language message as being from the service.

5. The method as recited in claim 2, further comprising the event message endpoint verifying type correctness of the data representation language message according to the data representation language schema subsequent to said receiving a message.

6. The method as recited in claim 2, wherein the data representation language schema defines a set of messages that the service may send to the event message endpoint, the method further comprising the event message endpoint verifying the correctness of the data representation language message from the service according to the data representation language schema.

7. The method as recited in claim 2, further comprising each of the one or more processes registering interest in one or more of the set of events generated by the service with the event message endpoint subsequent to said generating an event message endpoint.

8. The method as recited in claim 7,

wherein said registering interest in one or more of the set of events comprises each of the one or more processes providing an event handler callback method to the event message endpoint;

wherein said sending the data representation language representation of the event to one or more processes registered to receive the event from the service comprises:

the event message endpoint calling an event handler method of each
process registered with the event message endpoint to the event;
and

the event message endpoint passing the data representation language
representation of the event to each called event handler.

9. The method as recited in claim 7, further comprising:

a process unregistering interest in a first event of the service; and

the event message gate unsubscribing to the first event with the service
subsequent to said unregistering;

wherein the service is further configured to not send messages including data
representation language representations of the first event to event message
endpoints that are unsubscribed to the first event.

10. The method as recited in claim 2, further comprising receiving the data
representation language schema of the service in a service advertisement of the service.

11. The method as recited in claim 1, wherein the one or more processes are
executing within the client platform.

12. The method as recited in claim 1, wherein the event is a Java event.

13. The method as recited in claim 1, wherein said data representation
language is eXtensible Markup Language (XML).

14. A device, comprising:

a processor;

a memory coupled to said processor;

an event message gate unit configured to:

receive a message in a data representation language sent to the device in the distributed computing environment from a service in the distributed computing environment, wherein the message includes a data representation language representation of an event generated by the service; and

send the data representation language representation of the event to one or more processes registered to receive the event from the service.

15. The device as recited in claim 14, wherein the device is configured to:

receive a data representation language schema, wherein said data representation language schema defines a message interface for a set of events generated by the service; and

generate the event message gate unit according to the data representation language schema.

16. The device as recited in claim 15, wherein the event message gate unit is further configured to verify type correctness of the data representation language message according to the data representation language schema subsequent to said receiving a message.

17. The device as recited in claim 15, wherein the data representation

language schema defines a set of messages that the service may send to the event message gate unit, and wherein the event message gate unit is further configured to verify the correctness of the data representation language message from the service according to the data representation language schema.

18. The device as recited in claim 15, wherein the device is further configured to receive the data representation language schema of the service in a service advertisement of the service.

19. The device as recited in claim 14, wherein the event message gate unit is further configured to subscribe to one or more of the set of events generated by the service, and wherein the service is configured to send messages including data representation language representations of an event to subscribers to the event when the event is generated.

20. The device as recited in claim 14, wherein the data representation language message from the service includes an authentication credential for the service, wherein the event message gate unit is further configured to use the authentication credential for the service to authenticate the data representation language message as being from the service.

21. The device as recited in claim 14, wherein each of the one or more processes are configured to register interest in one or more of the set of events generated by the service with the event message gate unit subsequent to said generating an event message gate unit.

22. The device as recited in claim 21,

wherein, in said registering interest in one or more of the set of events, each of the one or more processes is configured to provide an event handler callback method to the event message gate unit;

wherein, in said sending the data representation language representation of the event to one or more processes registered to receive the event from the service, the event message gate unit is further configured to:

call an event handler method of each process registered with the event message gate unit to the event; and

pass the data representation language representation of the event to each called event handler.

23. The device as recited in claim 21,

wherein a first process is configured to unregister interest in a first event of the service;

wherein the event message gate is further configured to unsubscribe to the first event with the service subsequent to said unregistering; and

wherein the service is configured to not send messages including data representation language representations of the first event to event message gate units that are unsubscribed to the first event.

24. The device as recited in claim 14, wherein the one or more processes are executing within the client platform.

25. The device as recited in claim 14, wherein the event is a Java event.

26. The device as recited in claim 14, wherein said data representation language is eXtensible Markup Language (XML).

27. A device, comprising:

a processor;

a memory coupled to said processor;

a service process configured to:

generate an event;

generate a message in a data representation language, wherein the message includes a data representation language representation of the event generated by the service process; and

send the message to one or more event message gate units in the distributed computing environment;

wherein each of the one or more event message gate units are operable to distribute the data representation language representation of the event sent in the message from the service process to one or more processes registered to receive the event from the service process.

28. The device as recited in claim 27, wherein the device further comprises a service message gate unit, wherein said generating a message and said sending the message are performed by the service message gate unit on behalf of the service process.

29. The device as recited in claim 27, wherein the service process is further configured to:

provide a data representation language schema, wherein said data representation language schema defines a message interface for a set of events generated by the service; and

wherein the one or more event message gate units are generated according to the data representation language schema.

30. The device as recited in claim 29, wherein the data representation language schema defines a set of messages that the service may send to the event message gate units.

31. The device as recited in claim 29, wherein the service process is further configured to provide the data representation language schema in a service advertisement.

32. The device as recited in claim 27, wherein the service process is further configured to send messages including data representation language representations of an event to event message gate units subscribed to the event when the event is generated by the service process.

33. The device as recited in claim 27, wherein the service process is further configured to attach an authentication credential for the service to the data representation language message, wherein the authentication credential is configured for use in authenticating the data representation language message as being from the service process.

34. The device as recited in claim 27, wherein the events are Java events.

35. The device as recited in claim 27, wherein said data representation language is eXtensible Markup Language (XML).

36. A tangible computer readable medium comprising program instructions,

wherein the program instructions are computer-executable to implement:

receiving a message in a data representation language sent to a client platform in the distributed computing environment from a service in the distributed computing environment, wherein the message includes a data representation language representation of an event generated by the service; and

sending the data representation language representation of the event to one or more processes registered to receive the event from the service.

37. The tangible computer readable medium as recited in claim 36, wherein the program instructions are further computer-executable to implement:

receiving a data representation language schema on the client platform, wherein said data representation language schema defines a message interface for a set of events generated by the service; and

generating an event message endpoint for the client platform according to the data representation language schema, wherein said receiving a message and said sending the data representation language representation of the event to one or more processes are performed by the event message endpoint.

38. The tangible computer readable medium as recited in claim 37, wherein the program instructions are further computer-executable to implement the event message endpoint subscribing to one or more of the set of events generated by the service, wherein the service is configured to send messages including data representation language representations of an event to subscribers to the event when the event is generated.

39. The tangible computer readable medium as recited in claim 37, wherein the data representation language message from the service includes an authentication

credential for the service, wherein the program instructions are further computer-executable to implement the event message endpoint using the authentication credential for the service to authenticate the data representation language message as being from the service.

40. The tangible computer readable medium as recited in claim 37, wherein the program instructions are further computer-executable to implement the event message endpoint verifying type correctness of the data representation language message according to the data representation language schema subsequent to said receiving a message.

41. The tangible computer readable medium as recited in claim 37, wherein the data representation language schema defines a set of messages that the service may send to the event message endpoint, wherein the program instructions are further computer-executable to implement the event message endpoint verifying the correctness of the data representation language message from the service according to the data representation language schema.

42. The tangible computer readable medium as recited in claim 37, wherein the program instructions are further computer-executable to implement each of the one or more processes registering interest in one or more of the set of events generated by the service with the event message endpoint subsequent to said generating an event message endpoint.

43. The tangible computer readable medium as recited in claim 42,

wherein, in said registering interest in one or more of the set of events, the program instructions are further computer-executable to implement each of the one or more processes providing an event handler callback method to the event message endpoint;

wherein, in said sending the data representation language representation of the event to one or more processes registered to receive the event from the service, the program instructions are further computer-executable to implement:

the event message endpoint calling an event handler method of each process registered with the event message endpoint to the event;
and

the event message endpoint passing the data representation language representation of the event to each called event handler.

44. The tangible computer readable medium as recited in claim 42, wherein the program instructions are further computer-executable to implement:

a process unregistering interest in a first event of the service; and

the event message gate unsubscribing to the first event with the service subsequent to said unregistering;

wherein the service is further configured to not send messages including data representation language representations of the first event to event message endpoints that are unsubscribed to the first event.

45. The tangible computer readable medium as recited in claim 37, wherein the program instructions are further computer-executable to implement receiving the data representation language schema of the service in a service advertisement of the service.

46. The tangible computer readable medium as recited in claim 36, wherein the one or more processes are executing within the client platform.

47. The tangible computer readable medium as recited in claim 36, wherein the event is a Java event.

48. The tangible computer readable medium as recited in claim 36, wherein said data representation language is eXtensible Markup Language (XML).

X. EVIDENCE APPENDIX

No evidence submitted under 37 CFR §§ 1.130, 1.131 or 1.132 or otherwise entered by the Examiner is relied upon in this appeal.

XI. RELATED PROCEEDINGS APPENDIX

There are no related proceedings.